An Approximation Algorithm for the Two-Parallel Machines Scheduling Problem with Capacity Constraints *

Heng Yang
State Key Laboratory of Intelligent Technology and System
Department of Automation
Tsinghua University
Beijing, 100084, P.R.China

Yinyu Ye and Jiawei Zhang
Department of Management Sciences
Henry B. Tippie College of Business
The University of Iowa
Iowa City, Iowa 52242, USA

August 1, 2000

*Research supported in part by NSF grants DMI-9908077 and DMS-9703490.

Abstract

We consider the problem of scheduling n independent jobs on two identical parallel machines, with a limit on the number of jobs that can be assigned to a single machine, so as to minimize the total weighted completion time of the jobs. We study a semidefinite programming (SDP) based approximation algorithm for solving this problem and prove that the algorithm has a worst case ratio at most 1.1626.

Key words. Parallel machine scheduling, max-cut problem, semidefinite programming, approximation algorithm.

1 Introduction

In the parallel machine scheduling problem, we are given a set of n jobs that have to be scheduled on m identical parallel machines. Each job J_j $(j = 1, 2, \cdots, n)$ has a positive processing time p_j and a positive weight w_j. In a feasible schedule, every job J_j is processed for p_j time units on one of the m machines without interruption. Every machine can process at most one job at a time, and every job can be processed on at most one machine at a time. Let C_j denote the completion time of job J_j in a given schedule. Then, two well studied optimality criteria in the literature are: to minimize the makespan $C_{\max} = \max_{1 \leq j \leq n} C_j$, or to minimize the total weighted completion time $\sum_{1 \leq j \leq n} w_j C_j$. See Chen, Potts and Woeginger [3] and Hall [6] for excellent surveys on parallel machine scheduling problems.

Most of the parallel machine scheduling models assume that each machine can process arbitrarily number of jobs. However, in many flexible manufacturing systems, as well as VLSI chip production, it is important to balance the number of jobs assigned to the different production facilities. In [11], Tsai proposed the following two-parallel machines scheduling problem with capacity constraints. The model is the two-parallel machines scheduling problem with an upper bound $q(n/2 \leq q \leq n)$ on the number of jobs that can be processed on a single machine. Tsai developed a heuristic for this problem with the goal to minimize the makespan (we denote it by $P2/q/C_{\max}$), and showed that it is asymptotically optimal when jobs processing times are independent and uniformly distributed. For the same problem, Bramel, Chan and Simchi-Levi [1] showed that a modified version of the well-known Longest Processing Time First heuristic (LPT) has a worst case ratio of 7/6 and this bound is tight.

In this paper, we study the two-parallel machines scheduling problem with capacity constraints and our goal is to minimize the total weighted completion time. We will denote this problem by $P2/q/\sum w_j C_j$. Without the capacity constraint, the problem is NP-hard but admits a *fully polynomial time approximation scheme* [9]. We present here an 1.1626-approximation algorithm for the $P2/q/\sum w_j C_j$ problem with the capacity constraint.

2 Preliminaries

In this section, we show the relationship between the $P2/q/\sum w_j C_j$ problem and the $Max-(q, n-q)-Cut$ problem. In the $Max-(q, n-q)-Cut$ problem, we are given an undirected graph $G = (V, E)$ with vertex set $V = \{1, 2, \cdots, n\}$ and non-negative weights $w_{ij} = w_{ji}$ on the edges $(i,j) \in E$, to find a subset of vertices $S \subset V$ maximizes

$$\frac{1}{2} \sum_{(i,j)\in E, i\in S, j\in V\setminus S} w_{ij}$$

such that $n - q \leq |S| \leq q$, where $n = |V|$ and $n/2 \leq q \leq n$. The special case in which $q = \frac{1}{2}n$ is referred to as the Max-Bisection problem. The $Max - (q, n-q) - Cut$ problem can be formulated as follows:

$$\text{(MqC)} \quad \begin{aligned} &\text{Maximize} && \frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j) \\ &\text{subject to} && n - 2q \leq \sum_{j=1}^{n} x_j \leq 2q - n \\ & && x_j^2 = 1, \; j = 1, \ldots, n. \end{aligned}$$

We will show how any constant factor approximation algorithm for $Max - (q, n - q) - Cut$ problem translates into an approximation algorithm for $P2/q/\sum w_j C_j$ with constant performance guarantee. This idea was first proposed by Skutella [10], but we will use a modified rounding and improved analysis to get a tighter approximation bound.

We associate with each instance of the $P2/q/\sum w_j C_j$ problem a complete undirected graph $G = (V, E)$ in the following way: the vertex set $V = \{1, 2, \cdots, n\}$ corresponds to the job set $\{J_1, J_2, \cdots, J_n\}$; and the weight w_{ij} of the edge $(i,j) \in E$ is given by

$$w_{ij} = \min\{w_i p_j, p_i w_j\}.$$

The reason for this setting is, for any single machine scheduling problem with our objective function, the optimal schedule will assign the jobs with the order of nondecreasing p_j/w_j. Then, the partition $(S, V \setminus S)$ of the vertex set V can be interpreted as scheduling the n jobs on the two machines. And $|S| \leq q$ corresponds to the capacity constraints that each machine can process at most q jobs. Moreover, the total weighted completion time of a feasible schedule can be represented by the total weights of the edges with both endpoints in the same subset S or $V \setminus S$ plus a constant term $\sum_{1\leq j\leq n} w_j p_j$. One can verify

$$w^{TO} + \sum_{1\leq j\leq n} w_j p_j = \sum_{1\leq j\leq n} w_j C_j + w(S, V \setminus S), \tag{1}$$

where $w^{TO} = \sum_{1\leq i<j\leq n} w_{ij}$ is the total weights of all the edges of E; C_j is the completion time of job J_j in the optimal schedule corresponding to the partition

$(S, V \setminus S)$; and $w(S, V \setminus S)$ denotes the cut value of the partition $(S, V \setminus S)$, i.e.,

$$w(S, V \setminus S) = \frac{1}{2} \sum_{i \in S, j \in V \setminus S} w_{ij}.$$

Note that, for a given instance of the $P2/q/\sum w_j C_j$ problem, $w^{TO} + \sum_{1 \le j \le n} w_j p_j$ is a constant. Therefore, by (1), minimizing the total weighted completion time $\sum_{1 \le j \le n} w_j C_j$ of the schedule is equivalent to maximizing the cut value of $w(S, V \setminus S)$.

Let the minimal value of the $P2/q/\sum w_j C_j$ problem be Z^* and the maximal value of the corresponding $Max - (q, n-q) - Cut$ problem be w^*. Then, from (1) we have

$$Z^* + w^* = \sum_{1 \le j \le n} w_j C_j + w(S, V \setminus S) = w^{TO} + \sum_{1 \le j \le n} w_j p_j. \tag{2}$$

Skutella [10] proved the following lemma and theorem for the problem without capacity constraints, i.e., $P2//\sum w_j C_j$ or $P2/n/\sum w_j C_j$:

Lemma 1

$$Z^* \ge \frac{1}{2} w^{TO} + \frac{3}{4} \sum_{1 \le j \le n} w_j p_j. \tag{3}$$

Proof. See lemma 3.2 of [10]. ■

Theorem 1 *For any $\rho \le 1$, a ρ-approximation algorithm for $Max - Cut$ problem can be translated into an approximation algorithm for $P2//\sum w_j p_j$ with a performance guarantee $2 - \rho$.*

Remark. Note that (3) is also valid in our $P2/q/\sum w_j C_j$ problem. Since adding the capacity constraints will not decrease the minimal objective value. Also, $Max - Cut$ is a special case of $Max - (q, n-q) - Cut$ with $q = n$.

In general, the approximation bound ρ for $Max - (q, n-q) - Cut$ is a function of k, where $k = \frac{w^{TO}}{w^*}$ (It is clear that $k \ge 1$). Thus, in what follows, we will write $\rho(k)$ as ρ. We now show that the bound for $P2//\sum w_j p_j$ in the above theorem can be further improved.

Theorem 2 *For any $\rho(k) \le 1$, a $\rho(k)$-approximation algorithm for $Max - (q, n - q) - Cut$ problem can be translated into an approximation algorithm for $P2/q/\sum w_j p_j$ with a performance guarantee* $\min\{1 + \frac{1-\rho(k)}{3-k}, 2 - \rho(k)\}$.

Proof. By (3), we have

$$Z^* \geq \frac{1}{2}w^{TO} + \frac{3}{4}\sum_{1\leq j\leq n} w_j p_j = \frac{k}{2}\cdot w^* + \frac{3}{4}\sum_{1\leq j\leq n} w_j p_j. \tag{4}$$

Combining (3) and (1) we have

$$w^* \leq w^{TO} + \sum_{1\leq j\leq n} w_j p_j - Z^* \leq \frac{1}{2}w^{TO} + \frac{1}{4}\sum_{1\leq j\leq n} w_j p_j = \frac{k}{2}\cdot w^* + \frac{1}{4}\sum_{1\leq j\leq n} w_j p_j. \tag{5}$$

It follows that

$$(1-\frac{k}{2})w^* \leq \frac{1}{4}\sum_{1\leq j\leq n} w_j p_j. \tag{6}$$

Substitute (6) into (4) we have

$$Z^* \geq \frac{k}{2}\cdot w^* + 3(1-\frac{k}{2})\cdot w^* = (3-k)\cdot w^*. \tag{7}$$

And comparing (4) with (5), we see

$$Z^* \geq w^*. \tag{8}$$

Therefore, by (2),(7) and (8), if $w(S, V\setminus S) \geq \rho(k)w^*$, then

$$\begin{aligned}
\sum_{1\leq j\leq n} w_j C_j &= Z^* + w^* - w(S, V\setminus S) \\
&\leq Z^* + (1-\rho(k))\cdot w^* \\
&\leq Z^* + (1-\rho(k))\cdot \min\{\frac{1}{3-k}Z^*, Z^*\} \\
&= \min\{2-\rho(k), 1+\frac{1-\rho(k)}{3-k}\}\cdot Z^*
\end{aligned}$$

■

To get the performance guarantee of the algorithm, we should evaluate the maximal value of $\min\{1+\frac{1-\rho(k)}{3-k}, 2-\rho(k)\}$ for all possible $k \geq 1$. However, in the rest of this paper, we need only consider the case $1 \leq k \leq 2$. This is because, as we will see later, that $\rho(k) \geq \rho(2)$ for $k \geq 2$ in our algorithm; so that the maximal value will not attain at $k > 2$.

The advantage of placing k in the bound can be seen in the next section, where we consider the special cases: $q = n$—the Max-Cut case and $q = n/2$—the Max-Bisection case.

3 Two Special Cases

Before present our results, we introduce some notations. For a given $0 \leq \theta \leq 1$, let

$$a(\theta) = 1 - \frac{2}{\pi} \arcsin(\theta),$$

$$b(\theta) = \min_{-1 \leq y < 1} \frac{2}{\pi} \frac{\arcsin(\theta) - \arcsin(\theta y)}{1 - y},$$

and

$$c(\theta) = \min_{-1 \leq y < 1} \frac{1 - \frac{2}{\pi} \arcsin(\theta y)}{1 - y}.$$

Then, let

$$\alpha(\theta) = \max\{\frac{k}{2} a(\theta) + b(\theta),\ c(\theta)\}, \tag{9}$$

and

$$\beta(\theta) = \frac{n^2 - n}{4q(n-q)} a(\theta) + b(\theta). \tag{10}$$

From the lemma 3 of the next section, we know that there exists a algorithm for Max-Cut with performance guarantee $\rho(k) \geq \alpha(\theta)$ for any $\theta \in [0, 1]$. Then by Theorem 2, the algorithm can be translated into an approximation algorithm for $P2//\sum w_j C_j$ with a performance guarantee

$$1 + \frac{1 - \alpha(\theta)}{3 - k}.$$

If $1 \leq k \leq 1.515$, we let $\theta^* = 0.98$, where

$$1 + \frac{1 - \alpha(\theta^*)}{3 - k} \leq 1.07290.$$

If $1.515 \leq k \leq 2$, we let $\theta^* = 0.93$, where

$$1 + \frac{1 - \alpha(\theta^*)}{3 - k} \leq 1.07254.$$

Therefore, we have

Corollary 1 *There is a Max-Cut algorithm that can be translated into an approximation algorithm for* $P2//\sum w_j C_j$ *with a performance guarantee* 1.0729.

Note that only a 1.122-approximation algorithm for $P2//\sum w_jC_j$ was proved in [10].

It is also known from [12] and Lemma 3 of next section that the Max-Bisection possesses a $\rho(k)$-approximation algorithm, for any $\theta \in [0,1]$, where

$$\rho(k) \geq \frac{\alpha(\theta)}{1+\sqrt{1-\beta(\theta)}}.$$

Now, let us choose $\theta^* = 0.82$. Then, $a := a(\theta^*) = 0.3879467$, $b := b(\theta^*) = 0.5870386$, and $c := c(\theta^*) = 0.8042237$. Thus,

$$\alpha := \alpha(\theta^*) = \max\{\frac{k}{2}a + b,\ c\} \quad \text{and} \quad \beta := \beta(\theta^*) = (1-\frac{1}{n})a + b,$$

and

$$\rho(k) \geq \frac{\alpha}{1+\sqrt{1-\beta}}.$$

Now consider the maximal value

$$\max_{k\in[1,2]} 1 + \frac{1-\frac{\alpha}{1+\sqrt{1-\beta}}}{3-k}.$$

We see that the maximization occurs at two possible points such that $\frac{k}{2}a+b=c$ and $k=2$. In this case,

$$k = \frac{2(c-b)}{a}$$

yields the maximal value less than 1.1626, for n sufficiently large.

Corollary 2 *There is a Max-Bisection algorithm that can be translated into an approximation algorithm for $P2/\frac{n}{2}/\sum w_jC_j$ with a performance guarantee* 1.1626.

4 Approximation Algorithm for $Max\!-\!(q,n\!-\!q)\!-\!Cut$

Now consider general $q \geq \frac{n}{2}$. First we give the semidefinite programming relaxation of (MqC):

$$\text{(SDP)} \qquad \begin{aligned} w_{SDP} := \quad & \text{Maximize} \quad \frac{1}{2}\sum_{i<j} w_{ij}(1-X_{ij}) \\ & \text{subject to} \quad ee^T \bullet X \leq (2q-n)^2, \\ & \qquad\qquad\quad X_{jj} = 1, \quad j=1,...,n, \ X \succeq 0. \end{aligned} \qquad (11)$$

Here, the unknown $X \in R^{n\times n}$ is a symmetric matrix. Furthermore, $\bullet$ is the matrix inner product $Q \bullet X = \text{trace}(Q^T X)$, and $X \succeq Z$ means that $X - Z$ is positive semidefinite.

Obviously, (SDP) is a relaxation of (MqC), since for any feasible solution x of (MqC), $X = xx^T$ is feasible for (SDP); so that $w_{SDP} \geq w^*$.

The following approximation method is similar to the one in Frieze and Jerrum [4] and Ye [12], also see Nesterov [8] and Zwick [13].

Algorithm H

1. SDP Solving: Solve (SDP) to obtain the semidefinite matrix $\bar{X}$.

Repeat the following two steps for $[\epsilon^{-1}\log\epsilon^{-1}]$ times, where $\epsilon > 0$ is small constant as stated in Frieze and Jerrum [4], and output the best bisection.

2. Randomized Rounding: Generate a vector u from a multivariate normal distribution with 0 mean and covariance matrix $\theta^*\bar{X} + (1-\theta^*)I$, where I is the identity matrix. That is, generate

$$u \in N(0, \theta^*\bar{X} + (1-\theta^*)I),$$

then assign

$$\hat{x} = \text{sign}(u),$$

i.e.,

$$\hat{x}_i = \begin{cases} 1 & \text{if } u_i \geq 0 \\ -1 & \text{if } u_i < 0. \end{cases}.$$

Select the block $S = \{i : \hat{x}_i = 1\}$ or $S = \{i : \hat{x}_i = -1\}$ such that $|S| \geq n/2$. Let $\tilde{S} = S$.

3. Node Swapping: If $|\tilde{S}| \leq q$, stop. If $|\tilde{S}| > q$, for each $i \in \tilde{S}$, let $\zeta(i) = \sum_{j\in V\setminus\tilde{S}} w_{ij}$ and $\tilde{S} := \{i_1, i_2, ..., i_{|\tilde{S}|}\}$, where $\zeta(i_1) \geq \zeta(i_2) \geq ... \geq \zeta(i_{|\tilde{S}|})$. Then, remove node $i_{|\tilde{S}|}$ from $\tilde{S}$ and reassign $\tilde{S} := \{i_1, i_2, ..., i_{|\tilde{S}|-1}\}$. Repeat this swapping process till $|\tilde{S}| = q$. $\blacksquare$

Remark: As we have seen in the last section, the performance of the algorithm depends on both k and θ. We can choose the best θ for a specific k. Since we do not know k in advance, we can independently run the algorithm for $\theta \in \{0.01, 0.02, \cdots, 0.99, 1.00\}$ and then choose the best output.

The algorithm H can be derandomized; see Mahajan and Ramesh [7].

Lemma 2

$$w(\tilde{S}, V \setminus \tilde{S}) \geq \begin{cases} \frac{q}{|S|} \cdot w(S, V \setminus S) & \textit{if} \quad |S| \geq q, \\ w(S, V \setminus S) & \textit{otherwise.} \end{cases}$$

5 Analysis of the Approximation Method

In order to analyze the quality of bisection $\tilde{S}$, we define two random variables similar to those in [4, 12]:

$$w := w(S, V \setminus S) = \frac{1}{2} \sum_{i<j} w_{ij}(1 - \hat{x}_i \hat{x}_j) = \frac{1}{4} \sum_{i,j} w_{ij}(1 - \hat{x}_i \hat{x}_j)$$

and

$$m := |S|(n - |S|) = \frac{n^2}{4} - \frac{(e^T \hat{x})^2}{4} = \frac{1}{4} \sum_{i,j} (1 - \hat{x}_i \hat{x}_j).$$

Lemma 3 *For any given $\theta \in [0, 1]$, the approximation method H yields S satisfying the following two inequalities:*

$$E[w] \geq \alpha(\theta) \cdot w^*,$$

$$E[m] \geq \beta(\theta) \cdot q(n - q),$$

where $\alpha(\theta)$ and $\beta(\theta)$ are given by (9) and (10).

Proof. First, in [12] it was proved that

$$\mathrm{E}[w] \geq c(\theta) \cdot w^*.$$

Then, from Goemans and Williamson [5], Frieze and Jerrum [4], and Bertsimas and Ye [2], we have

$$\mathrm{E}[\hat{x}_i \hat{x}_j] = \frac{2}{\pi} \arcsin(\theta \bar{X}_{ij}), \quad i, j = 1, 2, \ldots, n. \tag{12}$$

Therefore,

$$\mathrm{E}[w] \ = \ \frac{1}{4} \sum_{i,j} w_{ij} \left(1 - \frac{2}{\pi} \arcsin(\theta \bar{X}_{ij})\right)$$

$$
\begin{aligned}
&= \frac{1}{4}\sum_{i,j} w_{ij}\left(1-\frac{2}{\pi}\arcsin(\theta)\right)+\frac{1}{4}\sum_{i,j} w_{ij}\left(\frac{2}{\pi}\arcsin(\theta)-\frac{2}{\pi}\arcsin(\theta\bar{X}_{ij})\right)\\
&\geq \frac{1}{2}(1-\frac{2}{\pi}\arcsin(\theta)w^{TO}+\frac{1}{4}\cdot b(\theta)\sum_{i,j} w_{ij}(1-\bar{X}_{ij})\\
&= \frac{k}{2}a(\theta)\cdot w^*+b(\theta)\cdot w_{SDP}\\
&\geq (\frac{k}{2}a(\theta)+b(\theta))\cdot w^*\\
&= \alpha(\theta)\cdot w^*.
\end{aligned}
$$

Finally, noting that

$$\sum_{i\neq j}\bar{x}_{ij}\leq (2q-n)^2-n,$$

we derive

$$
\begin{aligned}
n^2-\mathrm{E}[(e^T\hat{x})^2] &= \sum_{i\neq j}\left(1-\frac{2}{\pi}\arcsin(\theta\bar{x}_{ij})\right)\\
&\geq \sum_{i\neq j}\left(1-\frac{2}{\pi}\arcsin(\theta)+b(\theta)(1-\bar{x}_{ij})\right)\\
&\geq (n^2-n)\left(1-\frac{2}{\pi}\arcsin(\theta)\right)+4q(n-q)b(\theta)\\
&= \left(a(\theta)\frac{n^2-n}{4q(n-q)}+b(\theta)\right)\cdot 4q(n-q)\\
&= \beta(\theta)\cdot 4q(n-q).
\end{aligned}
$$

Thus, the second desired result in the lemma is obtained by the definition of m. ■

Let $t := q/n \in [1/2, 1]$ and define a new random variable

$$z(\gamma) := \frac{w}{w^*}+\gamma\frac{m}{q(n-q)}. \tag{13}$$

for a given $\gamma \in [0, \alpha/(\frac{1}{4t(1-t)}-\beta)]$. By lemma 2, we have

$$\mathrm{E}[z(\gamma)]\geq \alpha+\gamma\beta \quad \text{and} \quad z\leq k+\gamma.$$

Now we prove another lemma:

Lemma 4 *If random variable $z(\gamma)$ fulfills its expectation, i.e., $z(\gamma)\geq \alpha+\gamma\beta$, then*

$$w(\tilde{S}, V\setminus\tilde{S})\geq R^*\cdot w^*,$$

where

$$R^* = \min\{R_1(t,\gamma), R_2(t,\gamma)\},$$

$$R_1(t,\gamma) = \alpha + \beta\gamma - \frac{1}{4t(1-t)}\gamma,$$

and

$$R_2(t,\gamma) = 2\sqrt{\frac{(\alpha+\beta\gamma)\gamma t}{1-t}} - \frac{\gamma}{1-t}.$$

Proof. Suppose

$$w(S, V \setminus S) = \lambda w^* \quad \text{and} \quad |S| = \delta q = \delta t n \ (\geq n/2).$$

Note that

$$\lambda \geq \alpha + \beta\gamma - \frac{\delta t(1-\delta t)}{t(1-t)}.$$

There are two cases to analyze.

Case1. $\frac{1}{2t} \leq \delta \leq 1$. In this case

$$\begin{aligned} w(\tilde{S}, V \setminus \tilde{S}) &\geq w(S, V \setminus S) \\ &= \lambda w^* \\ &\geq (\alpha + \beta\gamma - \frac{\delta t(1-\delta t)}{t(1-t)}\gamma) \cdot w^* \\ &\geq (\alpha + \beta\gamma - \frac{1}{4t(1-t)}\gamma) \cdot w^* \\ &= R_1(t,\gamma) \cdot w^*. \end{aligned}$$

The last inequality holds since

$$\delta t(1-\delta t) \leq \frac{1}{4}.$$

Case 2. $1 \leq \delta \leq \frac{1}{t}$. In this case

$$\begin{aligned} w(\tilde{S}, V \setminus \tilde{S}) &\geq \frac{1}{\delta} w(S, V \setminus S) \\ &\geq \frac{1}{\delta}\lambda \cdot w^* \\ &\geq \frac{1}{\delta}(\alpha + \beta\gamma - \frac{\delta t(1-\delta t)}{t(1-t)}\gamma) \cdot w^* \end{aligned}$$

$$
\begin{aligned}
&= (\frac{\alpha+\beta\gamma}{\delta} + \frac{\gamma t}{1-t}\delta - \frac{\gamma}{1-t}) \cdot w^* \\
&\geq (2\sqrt{\frac{(\alpha+\beta\gamma)\gamma t}{1-t}} - \frac{\gamma}{1-t}) \cdot w^* \\
&= R_2(t,\gamma) \cdot w^*.
\end{aligned}
$$

■

Few remarks on functions $R_1(t,\gamma)$ and $R_2(t,\gamma)$. First, for a fixed t, $R_1(t,\gamma)$ is a linear function in γ and decreasing, and

$$R_1(t,0) = \alpha \geq 0 \quad \text{and} \quad R_1(t, \frac{\alpha}{\frac{1}{4t(1-t)} - \beta}) = 0.$$

Secondly, for a fixed t, $R_2(t,\gamma)$ is a concave function in $\gamma \in [0, \alpha/(\frac{1}{4t(1-t)} - \beta)]$, and

$$R_2(t,0) = 0 \quad \text{and} \quad R_2(t, \frac{\alpha}{\frac{1}{4t(1-t)} - \beta}) = 0.$$

6 The General Case

In this section, we fix $\theta = .82$. And we will show that the worst case of our algorithm is at $q = n/2$ or $t = 1/2$. Therefore, by the result in section 3, we get a $1.1626-$approximation algorithm for $P2/q/\sum w_j C_j$. We consider $t \in [1/2, 1]$ in two cases:

Case 1. $\frac{1}{2} \leq t \leq 0.54$. In this case, let

$$\gamma = \gamma_1 = \frac{\alpha}{2\beta}(\frac{1}{\sqrt{1-4\beta t(1-t)}} - 1), \tag{14}$$

where γ_1 maximizes $R_2(t,\gamma)$. Then

$$R_1(t,\gamma_1) = \alpha \cdot (\frac{1}{2} + \frac{1-\sqrt{1-a-4bt(1-t)}}{2a+8bt(1-t)})$$

and

$$R_2(t,\gamma_1) = \alpha \cdot \frac{2t}{1+\sqrt{1-a-4bt(1-t)}}.$$

And one can verify that $R_1(t,\gamma_1) \geq R_2(t,\gamma_1)$ (See Appendix). We let

$$\rho(k) = \min_{\frac{1}{2} \leq t \leq 0.54} R_2(t,\gamma_1),$$

and we conclude that the minimum of $R_2(t,\gamma_1)$ is obtained at $t = 1/2$ (See Appendix) and the minimum value is $0.8634 \cdot \alpha$.

Case 2. $0.54 \le t \le 1$. In this case, let

$$\gamma = \gamma_2 = \frac{\alpha}{2} \cdot \frac{1 - 4\beta t(1-t)}{2\beta^2 t(1-t) + \frac{(4t-1)^2}{8t(1-t)} + \beta(4t-1) - 8\beta t^2}. \tag{15}$$

Then

$$R_1(t,\gamma_2) = R_2(t,\gamma_2) =$$

$$\alpha \cdot \frac{8t(2t-1)(1-a-4bt(1-t))}{2(a+4bt(1-t)) \cdot (a + (4t-1-8t^2)) + 16b^2t^2(1-t)^2 + (4t-1)^2 - a^2}.$$

We let

$$\rho(k) = \min_{0.54 \le t \le 1} R_1(t,\gamma_2),$$

and the minimum of $R_1(t,\gamma_2)$ is obtained at $t = 0.54$ (See Appendix) and the minimum value is $0.9231 \cdot \alpha$.

In both cases, we have $R^* \ge \rho(k) \ge 0.8634 \cdot \alpha$. And the minimum is obtained at $t = 1/2$. Since α is independent of t, the performance guarantee of our algorithm for $P2/q/\sum w_j C_j$ is at most the one for $P2/\frac{n}{2}/\sum w_j C_j$, when $\theta = 0.82$.

Theorem 3 *Algorithm H is a 1.1626-approximation algorithm for $P2/q/\sum w_j C_j$ problem.*

Note that for $n/2 < q < n$, the performance guarantee of Algorithm H can be strictly between 1.0729 and 1.1626, which correspond to the cases $q = n$ and $q = n/2$ respectively, by selecting a suitable θ,

7 Appendix

Lemma 5 *If $\frac{1}{2} \le t \le 0.54$ then $R_1(t,\gamma_1) \ge R_2(t,\gamma_1)$.*

Proof. We will show that for $\frac{1}{2} \le t \le 0.54$,

$$\frac{1}{2} + \frac{1 - \sqrt{1-a-4bt(1-t)}}{2a + 8bt(1-t)} \ge \frac{2t}{1 + \sqrt{1-a-4bt(1-t)}}.$$

That is,

$$\frac{1+\sqrt{1-a-4bt(1-t)}}{2}+\frac{1-(1-a-4bt(1-t))}{2a+8bt(1-t)}\geq 2t$$

or

$$1+\frac{\sqrt{1-a-4bt(1-t)}}{2}\geq 2t.$$

Thus, we only need to prove

$$(4b-16)t^2-(4b-16)t-(3+a)\geq 0.$$

The inequality holds for $0.4572 \leq t \leq 0.5428$ when $a = 0.387947$ and $b = 0.5870386$. ■

Lemma 6 *For* $\frac{1}{2} \leq t \leq 0.54$, *the function*

$$f(t)=\frac{2t}{1+\sqrt{1-a-4bt(1-t)}}$$

obtained it's minimum at $t = \frac{1}{2}$.

Proof. Let $g(t) = 1/f(t)$. We have

$$g^{'}(t)=-\frac{1}{2}t^{-2}-\frac{1}{2}t^{-2}(1-a-4bt(1-t))^{\frac{1}{2}}+\frac{1}{4}t^{-1}(1-a-4bt(1-t))^{-\frac{1}{2}}(-4b+8bt).$$

Let $g^{'}(t) = 0$ we get

$$1+\sqrt{1-a-4bt(1-t)}=\frac{2bt(2t-1)}{\sqrt{1-a-4bt(1-t)}},$$

i.e.,

$$4b(b-1)t^2+4abt-a(1-a)=0.$$

There is no real root for this equation. Let $t = 0$ the equation value < 0. Then $f(t)$ is increase function. ■

Lemma 7 *For* $0.54 \leq t \leq 1$, *the function*

$$f(t)=\frac{8t(2t-1)(1-a-4bt(1-t))}{2(a+4bt(1-t))\cdot(a+(4t-1-8t^2))+16b^2t^2(1-t)^2+(4t-1)^2-a^2}$$

obtains it's minimum at $t = 0.54$.

Proof. Note that

$$f(t) = \frac{-8t(2t-1)}{a + 4bt - 16t^2 - 4bt^2 + 8t - 1}.$$

Then

$$f^{'}(t) = -8 \cdot \frac{4bt^2 + (-4+4a)t + 1 - a}{((-16-4b)t^2 + (8+4b)t + a - 1)^2}.$$

Let $f^{'}(t) = 0$, i.e.,

$$4bt^2 + (-4+4a)t + 1 - a = 0.$$

The roots are 0.626817 and 0.415863. We conclude that the minimum of $f(t)(0.54 \leq t \leq 1)$ obtained it's minimum at either $t = 0.54$ or $t = 1$. By comparing these two values, we get that the minimum is obtained at $t = 0.54$. ■

References

[1] J. Bramel, L. Chan and D. Simchi-Levi, "Worst-Case Analysis of the Modified LPT Heuristic for the Two-Machine Makespan Problem with Capacity Constraints," submitted to *Discrete Applied Mathematics.*

[2] D. Bertsimas and Y. Ye, "Semidefinite relaxations, multivariate normal distributions, and order statistics," *Handbook of Combinatorial Optimization (Vol. 3)*, D.-Z. Du and P.M. Pardalos (Eds.) pp. 1-19, (1998 Kluwer Academic Publishers).

[3] B. Chen, C. N. Potts and G. J. Woeginger, "A Review of Machine Scheduling: Complexity, Algorithms and Approximability," *Handbook of Combinatorial Optimization (Vol. 3)*, D.-Z. Du and P.M. Pardalos (Eds.) pp. 21-170, (1998 Kluwer Academic Publishers).

[4] A. Frieze and M. Jerrum, "Improved approximation algorithms for max k-cut and max bisection," *Proc. 4th IPCO Conference* (1995) 1-13.

[5] M. X. Goemans and D. P. Williamson, " Improved approximation algorithms for Maximum Cut and Satisfiability problems using semidefinite programming," *Journal of ACM* 42 (1995) 1115–1145.

[6] L. A. Hall, "Approximation Algorithm for Scheduling," in *Approximation Algorithms for NP-hard Problems*, D. S. Hochbaum (Eds.) pp 1-45, (1996 PWS Publishers).

[7] S. Mahajan and H. Ramesh, "Derandomizing semidefinite programming based approximation algorithms," in *36th FOCS*, (1995) pp. 162-169.

[8] Y. E. Nesterov, "Semidefinite relaxation and nonconvex quadratic optimization," *Optimization Methods and Software* 9 (1998) 141-160.

[9] S. Sahni, "Algorithms for Scheduling Independent Tasks," *Journal of ACM*, 23 (1976) 116-127.

[10] M. Skutella, "Semidefinite Relaxations for Parallel Machine Scheduling," in *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, (1998) pp. 472-481.

[11] L. H. Tsai, "Asymptotic Analysis of an Algorithm for Balanced Parallel Processor Scheduling," *SIAM Journal on Computing*, 21, (1992) pp. 59-94.

[12] Y. Ye, "A .699-approximation algorithm for Max-Bisection", *Manuscript*, March 1999.

[13] U. Zwick, "Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to Max-Cut and other problems," in *31th STOC* (1999).